Square Wave Generator and its use in Digital Spirit Level

Alvin Jorgih

Georgia Institute of Technology
School of Electrical and Computer Engineering

Submitted
November 25, 2020

Introduction

This document outlines the functionality of a square wave generator that our team has completed and the application of the device. The peripheral can generate a continuous 50% duty cycle square wave at a given duration. Our prioritized features include playing a note for a given duration, stop playing the note, and playing the note continuously. The final product met our proposed specifications and intentions with additional features that the team explicitly offered.

Device Functionality

To use our peripheral the user sends a 16-bit binary value to the address 0x00F0. The ten most significant bits are used as half the period of the square wave that is desired. The device will count up to that value using a 100 kHz clock and then flip the output. The six least significant bits represent the duration of the square wave. The device will count up to that value using a 10 Hz clock then stop playing the square wave. If the user sends a duration “0”, it will play a note indefinitely until another out is received. Take caution not to use numbers greater than 1023 for the frequency value and greater than 63 for duration value. The user should also delay their program for the duration of the peripherals square wave.

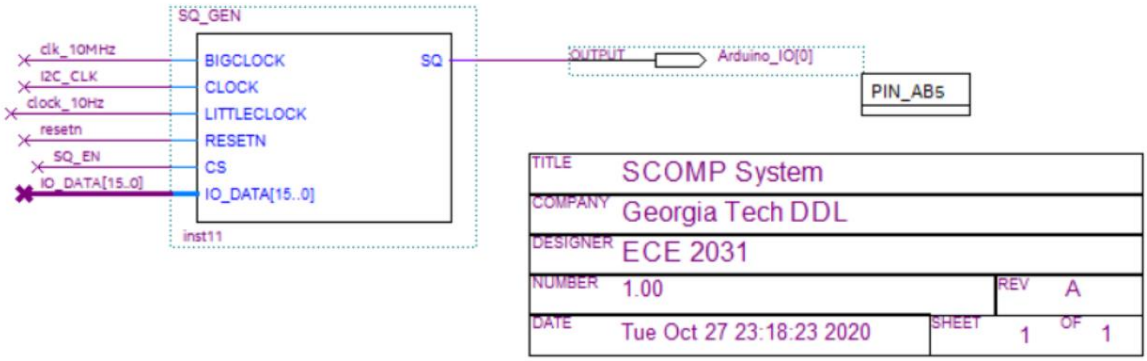

Figure 1. Square wave generator peripheral will count up to the value received from SCOMP using that as half a period of a square wave.

Digital Spirit Level Functionality

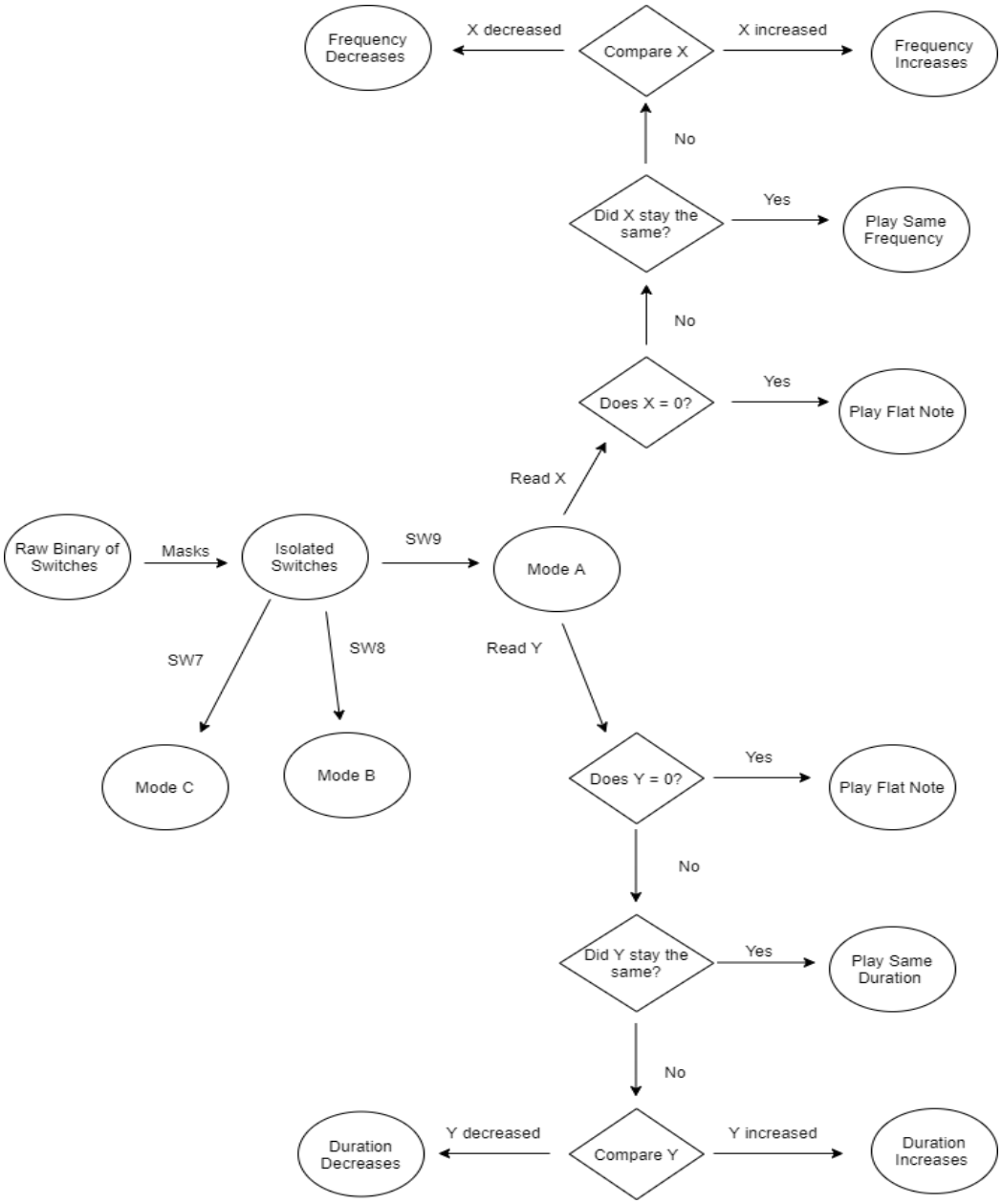

Figure 2. Flow diagram showing the functionality of our applications including accessing different game modes, instructions to use the switches and accelerometer.

The digital spirit level uses the DE10-Lite Board's accelerometer to create a orientation-dependent output. The spirit level offers different game modes activated by flipping specific switches. The accelerometer reads X acceleration values and Y acceleration values demonstrating application functionality; tilting the board farther in the X direction fluctuates the frequency of the note while tilting the board farther in the Y direction fluctuates the duration of the note.

Design Decisions

With the state machine, the user can stop playing a note and produce no sound. This allows the user to play complex music, communicate multi-faceted information, and avoid indefinite notes. For peripheral design, the lowest 6 bits are used for the duration and the highest 10 bits for frequency. By using 10 bits, the user can play a "library" of pre-calculated frequency values. They can then load these values and add the desired duration value allowing for rapid code modifications. Due to how the I/O bus was split, the device is capable to play a note of a max duration of 6.3s and the minimum being 0.1s. For frequency, there is a range of 48Hz – 100kHz. Though it is important to note the average human range about 30Hz – 19kHz, so only very low frequencies will be missing.

Conclusion

Our team successfully created an I/O peripheral for SCOMP that generated 50% duty cycle square waves from $C_2 - C_7$. Our application uses the accelerometer as inputs to demonstrate fluctuations in the frequency and duration of the outputted note. The application also uses switches to choose from different game modes, the user to select from. In retrospect, attempting to propose a solution for the limitations of this device, which in this case the device can only play limited frequency and a limited amount of time. It would be ideal to optimize this peripheral by increasing the number of bits the peripheral receives or making the first six significant bits as the duration and the remaining ten bits as the frequency to resolve the limitations.